Chapter 6

Risk Management

Contents

6.1 Risk Management: An Investment in Success

"Software is so vital to military systems that, without it, most could not operate at all. Its importance to overall system performance, and the generally accepted notion that software is always inadequate, makes software the highest risk item and must be steadfastly managed. Failure to identify and address risk has been the downfall of many DoD acquisition programs. The system component with the greatest inherent risk has historically been software." — Lloyd K. Mosemann, II (Deputy Assistant Secretary of the Air Force) [MOSEMANN95]

Software acquisition and development are two of the most risk prone challenges of this decade. Risk factors are always present that can negatively impact the development, acquisition, or maintenance processes. If neglected, these factors can tumble an unwitting program manager into acquisition disaster. To triumph in software acquisitions you must actively assess, control, and reduce software risk on a routine basis.

In Chapter 2, *Software Acquisition Success: Exception or Rule?*, poor management is cited as the chief reason why software acquisitions fail. Our software acquisition, development, and maintenance processes are too often immature, chaotic, and unpredictable. Our estimates of software cost, schedule, size, and complexity are inadequate. Our problem solving and decision making processes are poor because we do not plan, measure, track, or control the process and product. Risk management addresses all these shortcomings.

All DoD acquisition program managers (PMs) are responsible for establishing and executing a risk management program that satisfies the policies contained in **DoDD 5000.1**. Program-unique requirements or circumstances must be balanced within proven risk management principles and practices. PMs must take an active role in the risk management process and ensure it leads to a judicious use of program resources. Past DoD practices generally treated risk management as a systems engineering or cost-estimating technique, or possibly an independent function distinct from other program management activities. Today, risk management is recognized as a vital management tool that spans the entire acquisition life cycle. It addresses and interrelates the sources of acquisition failure — cost, schedule, and performance.

6.1.1 Integrated Risk Management

This chapter addresses risk management from two perspectives — the buyer (Government) and the supplier (software developer or maintainer). To be a successful buyer, you must have a thorough understanding of the supplier's risk as well as your own. Effective acquisition risk management depends on the ability to integrate the contractor(s) risk management process with your risk management program and the Integrated Product Team (IPT) process.

6.1.2 Risk Management Return on Investment

Risk management is *Number One* of all the software acquisition best practices discussed throughout these Guidelines. Like those that follow, risk management is a necessary activity that requires time and funding investment. Program managers often complain that they are required to integrate so many best practices in their programs. But how are they to pay for them? Risk management is one of those sticky investments that if you choose not to pay now, you will pay much more later. Properly performed, risk management becomes a mindset, a way of life, a management style. It is always present and plays a leading role in all decision making. Because it was seldom performed in the past, DoD acquisition reform initiatives are highlighting its practice. Once it becomes ingrained in our acquisition and management processes, this discussion will only need to address the how — not the why.

Risk management saves money and improves program and product quality with a relatively low investment. Hall explains that risk management return on investment (ROI) is calculated by adding the savings gained by managing risks, then dividing by the cost to manage them, as expressed in the following equation.

$$\mathbf{ROI_{(RM)}} = \frac{\sum \mathbf{Savings}}{\text{Cost}}$$

Risk management costs include the total investment in resources:

- Time spent in meetings,
- Cost of reporting and documenting risk information,
- Staff to develop and implement risk plans.

Risk management savings includes cost avoidance and cost reductions.

- **Cost avoidance** is the difference between possible cost without risk resolution and actual cost with resolution.
- **Cost reduction** is the difference between planned and actual costs.

According to Hall, if savings are accrued by cost reductions, it is possible to under-run the budget. Risk management activities can also uncover opportunities for the program to perform better than the baseline plan. [HALL97]

According to Charette, cultivating a disciplined risk management process can result in 50% or more in productivity gains, and greatly increase the potential for producing a quality product. [CHARETTE89] Without effective risk management, Norman Augustine’s Law of Counter-Productivity takes over, where “*It costs a lot to build bad products.*” [AUGUSTINE83]

While doing nothing to minimize risk can prove costly or disastrous for a program, the cost to avert all risk can be exorbitantly high. The costs in time, money, and effort should be balanced to provide minimized (acceptable) risks and not become a major drain on the project itself. This optimum benefit point is shown in Figure 6-1.

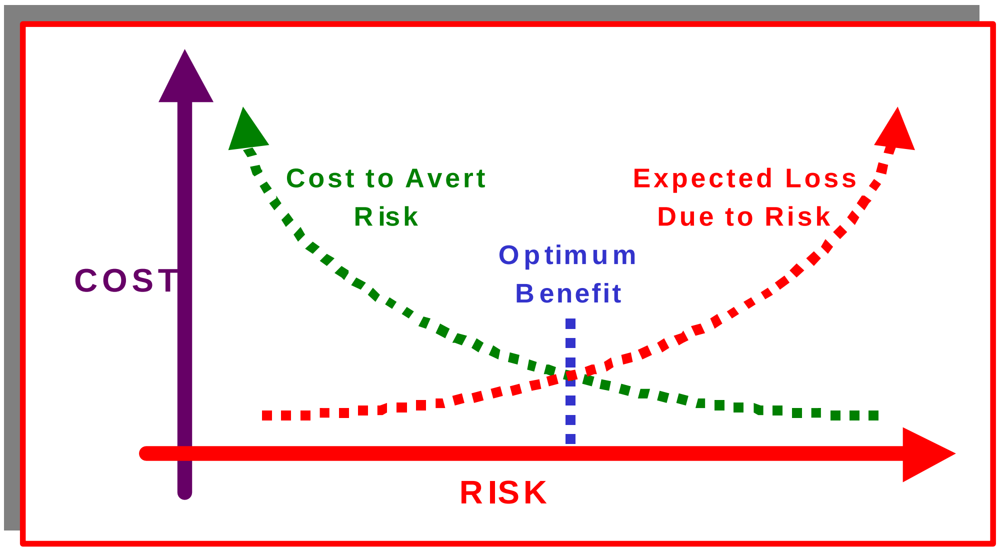


Figure 6-1. Cost / Benefits of Effective Risk Management [CHARETTE89]

6.1.3 Risk Management Reserve of Funds

"Why is it that most projects are functionally and/or calendar late, and so few on schedule? Consider that most projects never tried any risk management; therefore, their plans accounted only for the work recognized as absolutely necessary to build the software. These projects maintained no significant contingency fund for dealing with all the risks that might or might not manifest as actual problems. Why do we need such a contingency fund? Because no project ever runs exactly to plan!" — Tim Lister [LISTER97]

A reserve to meet the needs of contingencies must be budgeted into your program. Risk management must be a quantifiable effort, and the ROI for should be calculated and tracked. To justify your efforts and record your successes you must establish a database of costs to mitigate risks versus the potential costs of not using risk management.

The FY97 **Report of the Quadrennial Defense Review (QDR)** explains that complex, advanced technology acquisition programs all bear some risk of costing more than planned. When unforeseeable growth in costs occurs, offsets from other programs must be found, which disrupts overall DoD modernization programs. The QDR states that each military department must establish a prudent funding reserve in its out-year plans to offset these types of cost increases and significantly reduce the destabilizing risk factors affecting our modernization programs.

6.2 Risk Management Fundamentals

NOTE: The remainder of this chapter should be read in conjunction with "Risk Management Guide for DoD Acquisition" published by the Defense Acquisition Management College (DSMC) Press, Ft. Belvoire, VA. Version 2.0 is available on the Defense Acquisition Deskbook CD or website.

Risk is defined as the probability of an undesirable event occurring and the impact of that event occurring. A risks is the precursor to a problem. It is the probability that, at any given point in the system life cycle, its planned goals will not be achieved within available resources. There is a high probability that you will have less than a full understanding of the requirements of either the software product or the process before development begins. You also run the risk that it will take longer and cost more than expected. Trying to totally eliminate risk is a futile endeavor — however, *managing risk is something you can and must do*. To know whether an event is truly "*risky*," you must have an understanding of the potential consequences of the occurrence/nonoccurrence of that event. Software-intensive systems acquisitions risk failure in four ways (and combinations thereof).

1. The product that was ordered is not the product the end user wanted.
2. The product does not meet performance requirements (operationally or logistically).
3. Actual costs are higher than budgeted.
4. Delivery of the product is too late.

Risk management is a systematic approach to identifying, analyzing, and controlling events with a potential for causing unwanted change. It is through the risk management process that program risks are assessed and systematically managed to an acceptable level. To be an effective manager, you must identify and mitigate risks throughout the system life cycle. You will be hard pressed to eliminate all risks, but should act on those risks most critical to program success to the point where they become manageable. Implementing risk management (based on a judicious mix of structured processes, practical methods, and *common sense*) gives you a greater chance to succeed and to straighten things out while problems are still solvable.

6.2.1 Risk Management Team

It is important to form an effective government/ industry risk management team at the onset of the acquisition. Sharing information and concerns, careful listening, and timely responses among mutually bound partners are essential risk management activities. Industry's belief that their concerns about risk will be addressed by the Government is vital. Conversely, the System Program Office's (SPO) ability to rely on its industry partner(s) to provide quality solutions within user requirement parameters enhances the probability for risk management success.

The SEI **Team Risk Management (TRM)** views the joint government/industry team approach as a continuing process throughout the acquisition and development life cycle. The objective of TRM is to create a non-threatening atmosphere for risk control and mitigation. With TRM, plans are put in place to control risks while they are still manageable. Under this method, the Government and contractor perform risk management together. The seven principles of TRM are illustrated in Table 6-1.

Principle	Effective Risk Management Requires:
Shared Product Vision	Sharing product vision based upon common purpose, shared ownership, and collective commitment Focusing on results
Teamwork	Working cooperatively to achieve a common goal Pooling talent, skills, and knowledge
Global Perspective	Viewing software development within the context of the larger systems-level definition, design, and development Recognizing both the potential value of opportunity and the potential impact of adverse effects
Forward-Looking View	Thinking toward tomorrow, identifying uncertainties, anticipating potential outcomes Managing program resources and activities while anticipating uncertainties
Open Communication	Encouraging free-flowing information at and between all project levels Enabling formal, informal, and impromptu communication Using consensus-based processes between customer and supplier that value the individual voice
Integrated management	Making risk management an integral and vital part of program management Adapting risk management methods and tools to a project's infrastructure and culture
Continuous process	Sustaining constant vigilance Identifying and managing risks routinely throughout all phases of the project's life cycle

Table 6-1. Principles of Team Risk Management [HIGUERA94]

Table 6-2 highlights the advantages of Team Risk Management and identifies the commitment required by the team (customer and supplier) to achieve the advantage.

Advantage	Description	Required Commitment
Improved Communications	• The aspect of routine communications includes both customer and supplier. Risks are treated by all as depersonalized issues that threaten the common goal of a successful program. • By openly sharing risks, both the customer and supplier are able to draw on each other’s resources in mitigating risks and enabling rapid response to developing risks or problems.	• Move beyond finger pointing and resolve project risks as a joint responsibility. • Encourage all forms of communications (e.g., telephone and electronic mail) among all team members. • Encourage all to explore what could cause the program to go off track. • Allow for more meetings and more travel initially.
Multiple Perspectives	• Team members are not limited to looking for mitigation strategies among their own limited areas of control. • Bringing both customer and supplier together in mitigating risks opens doors to strategies that both can do together, but that neither could do alone.	• Accept the philosophy that the team can arrive at better solutions that any individual – even the program manager – can alone.
Broader Base of Expertise	• The combination of customer and supplier brings together a richer pool of experience in perceiving and dealing with risks. • The customer often brings better perspectives on the application domain and “what’s possible to change.” • The supplier often brings better perspectives on the technical domain and “what’s possible to do.”	• Accept all the unique perspectives that others bring to the table.
Broad-Based Buy-In	• Risks and mitigation strategies are cooperatively determined by the team (customer and suppler), so all accept the results of the process. “Second guessing” and criticism after the fact are eliminated. • Over time, trust develops and expectations are realized. This paves the way for strengthened relationships and the power of teamwork.	• Encourage and allow teams to meet, discuss, and agree. • Invest in improving meeting skills. • Use outside facilitation as required.
Risk Consolidation	• Structured methods bring together risks identified in each organization, giving decision makers a more global perspective and highlighting areas of common interest and concern.	• Accept that risk is inherent in enterprise. • Abandon the notion that risks should not be discussed until a mitigation strategy has been identified.

Table 6-2. Team Risk Management Benefits [HIGUERA94]

ATTENTION: Although it is desirable to have provisions for TRM as part of the contract, the SEI has concentrated their work on building the team environment after contract award and the team is in place.

6.2.2 Risk Management Process

Risk management costs time and money. However, *it is always less expensive to be aware of and deal with risks than to respond to unexpected problems*. It is your responsibility to put a process in place that enables you and your team to identify, analyze, plan, track, and relentlessly control risk. A risk that has been analyzed and resolved ahead of time is much easier with which to deal than a problem that surfaces unexpectedly. [BLUM92]

Risk management includes continuously assessing what can go wrong in programs (what the risks are), determining which risks are the most critical (risk assessment and prioritization), and implementing strategies to deal with risks (risk mitigation). In addition to relative importance, it is essential to quantify the impact of the risk if it occurs. This provides a benchmark against which to compare the cost of mitigating the risk. Risk can be quantified in dollar or schedule terms. If the risk occurrence would result in a $1,000,000 impact, you would probably be justified in spending $100,000 to mitigate it. Risk management normally goes through the generic, sequentially functions listed in Table 6-1. Risk management also occurs continuously, concurrently, and iteratively throughout the system life cycle (e.g., planning for one risk may identify another). [HIGUERA96] Figure 6-2 illustrates the Risk Management Process.

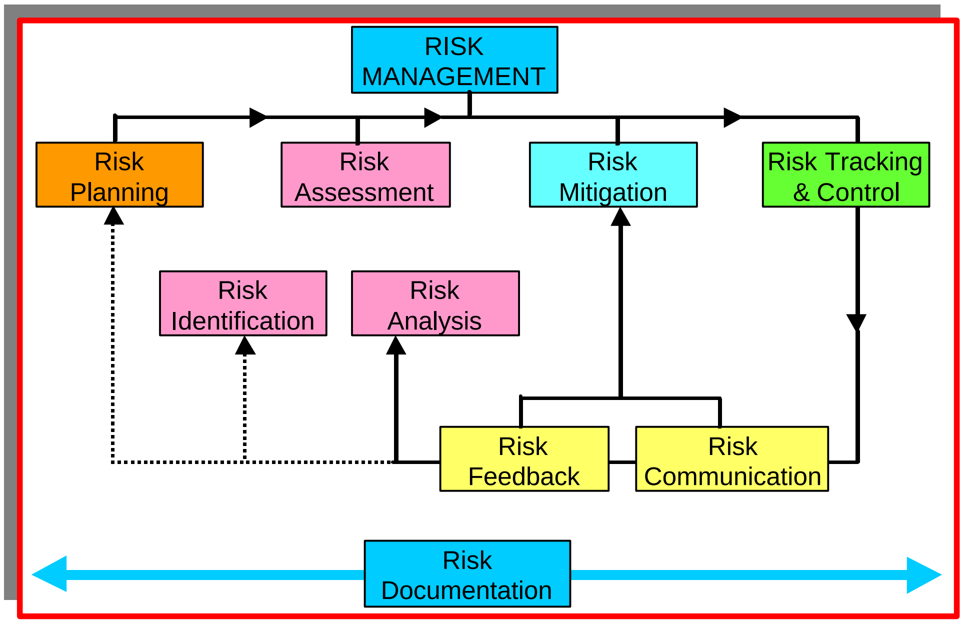


Figure 6-2. Risk Management Process [CONROW97]

NOTE - There are other risk management processes, both formal and informal. In the absence of one you know to be applicable and proven, we suggest this process.

6.2.3 Risk Management Planning

Risk planning is the process of developing and documenting an organized, comprehensive, and interactive strategy and methods for identifying and tracking risk areas, developing risk-mitigation plans, performing continuous risk assessments to determine how risks have changed, and planning adequate resources. These activities are illustrated in Figure 6-3.

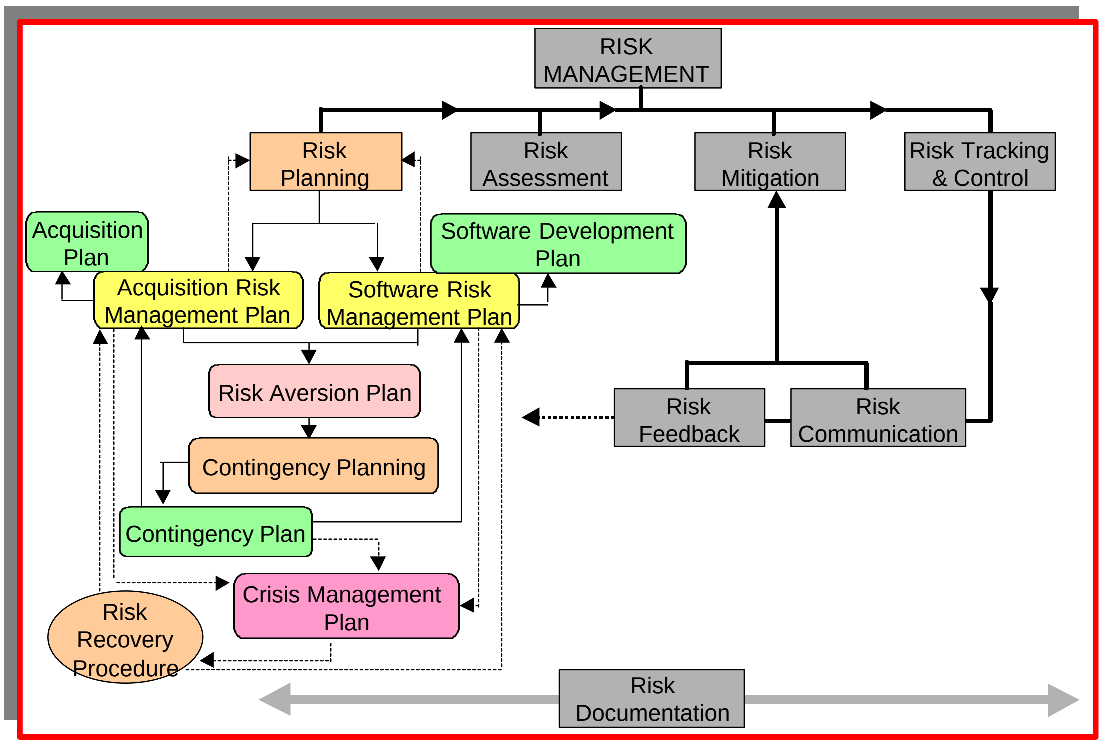


Figure 6-3. Risk Management Planning

It is important to be aware of your software risk factors. It is also important to know about risk management methods (discussed below). However, the real key to software risk management is planning and implementing your plan. Risk management planning is an on-going effort to identify all significant risk probabilities and to minimize their occurrence and/or impact. *Planning prevents most risks from becoming problems*. If you do not know your risks and their potential impacts, you are planning to let risks fester. Furthermore, the events used as decision points in a contracted effort will have greater meaning if they correspond to those points when important program uncertainties become known. Good management not only entails reasonable estimation, it means leaving nothing to chance (or at risk) that you can control through planning. Keeping an historical perspective on what has and has not worked for programs similar to yours is vital. East Roman Emperor and general, Maurice, having defeated the Persians and Avars circa AD 600, said the following.

> *"In battles and in every action against the enemy, the wise general, even the most courageous, will keep in mind the possibility of failure and defeat and will plan for them as actually occurring…The sharp general takes into account not only probable dangers, but also those which may be totally unexpected."* [MAURICE600AD]

6.2.3.1 Risk Management Plan

The Risk Management Plan (RMP) is a controlling document that states how risk analysis and procedures will be applied to your program. It describes all aspects of the risk identification, assessment, tracking, and control process.

The RMP consists of an integrated approach to each risk item identified. It should be based on answering the standard questions of *why*, *what*, *when*, *who*, *where*, *how*, and *how much*. The acquisition RMP must be thoroughly integrated into the Acquisition Strategy and Plan. Likewise, the software development RMP must be integrated into the Software Development Plan (SDP). For example , if the developer plans to buy information by building a prototype on fault-tolerant features, that same prototype may be used to reduce latent software defects or to involve the user in interface design. If the need for a 10-week prototype development and demonstration period has been identified, that time must also be added to the overall development schedule to keep it current and realistic.

The following is an outline of a typical risk management plan:

1. Scope of the document
2. Objective of the risk management program
3. Organization
 3.1. Who
 3.2. Responsibilities
 3.3. Identify additional technical expertise needed
 3.4. Reference the project's risk management process and procedures
4. Risk Assessment
 4.1. Risk Estimate of the Situation (RES)
 4.2. Risk Identification
  4.2.1. Reference the checklist to be used to help identify risks
  4.2.2. Define the risk taxonomy to be used for this program or project
 4.3. Risk Analysis
5. Risk Mitigation
6. Risk Tracking and Control
 6.1. Identify risk tracking indicators and measurements
 6.2. Identify the individuals or groups to receive the risk information
 6.3. Determine the review and reporting schedule
7. Risk Communication
8. Risk Feedback
9. Risk Documentation
 9.1. Identify documentation needs
 9.2. Specify documentation format
 9.3. Identify the repository for the documents

Note that the risk management plan (or any other plan, for that matter) should not contain processes or procedures. Processes and procedures should be common elements shared among the projects within a program and across programs. The plan should be a specific tailored implementation of the processes and procedures for a specific project or program and should reference the generic processes and procedures.

6.2.3.2 Completing the Plan

The risk planning phase is not finished until all elements of the plan have been addressed. The following sections contain examples of information you may want to include in your plan.

(1.0) Scope of the Document
Describe the system or subsystem being acquired or developed. Identify the organizations that will be using the risk management plan.

(2.0) Objective
Describe the desired outcome of successful implementation of the plan.

(3.0) Organization

(3.1) Who
Identify by position who will participate in the plan. For a small project or program, one individual working part time may be sufficient to manage the risk effort. However, for larger projects or programs, a small staff may be required. Don’t forget to include management.

(3.2) Responsibilities
You should specify the specific responsibilities of each individual in this portion of the plan. Some projects/programs have found it beneficial to identify a “risk officer” who has the specific responsibility of ensuring risks are monitored, issues are identified, and mitigation strategies are carried out.

(3.3) Identify additional technical expertise needed
Your organization may not have all the technical expertise required. If this is the case, identify any expertise needed but not available.

(3.4) Reference the project’s risk management processes and procedures
These processes and procedures should be those used within the program or organization with specific tailoring for the project described in this risk management plan.

(4.0) Risk assessment
Risk assessment is the process of identifying and analyzing program areas and critical technical process risks to increase the likelihood of meeting performance, schedule, and cost objectives. It includes the following activities:

- **Risk Estimate of the Situation (RES)**
- **Risk Identification**
- **Risk Analysis**

(4.1) Risk Estimate of the Situation
The initial part of risk assessment includes performing a Risk Estimate of the Situation. [NAVY95] As outlined in the **Defense Acquisition Deskbook**, the RES clearly identifies four major acquisition program elements.

- **Objectives**. Measurable and controllable acquisition goals.
- **Strategies**. Broad constraints or rules under which acquisition goals can be met.
- **Actions**. What happens during a given situation.
- **Resources**. Resources constrain acquisition objectives, strategies, and actions.

Grouping these elements together aids in determining variables in the environment (both technical and non-technical) in which the system is to operate. Early identification of false efforts and quick recognition of missing ones (both precursors to schedule delays and cost overruns) are primary RES goals. The RES helps us understand how each element interacts with and affects other elements so we can determine how they contribute to overall program success criteria. [CHARETTE89]

(4.2) Risk Identification

Risk identification is the process of examining the program areas and each technical process to identify and document the associated risk. One method of identifying risks is to use a checklist or taxonomy (i.e., a division of items into ordered groups or categories) that identify potential risk factors. Checklists or a taxonomy can be useful in ensuring risks or categories of risks are not overlooked during risk identification.

(4.2.1) Risk Checklists

Reference the checklists to be used in identifying project risks. Table 6-3 illustrates Boehm's Top-Ten Risk Identification Checklist. The checklist identifies the top-ten risk factors that can endanger a software program's quality, cost, and schedule goals. It also provides a set of corresponding risk management techniques proven to be successful in avoiding or resolving each particular source of risk.

Risk Item	Risk Management Techniques
Personnel shortfalls	Staffing with top talent; job matching; team building; morale building; cross-training; pre-scheduling key people
Unrealistic schedules and budgets	Detailed, multi-source cost and schedule estimation; design-to-cost; incremental development; software reuse; requirements scrubbing
Developing the wrong software functions	Organizational analysis; mission analysis; operations concept formulation; user surveys prototyping; early user's manuals
Developing the wrong user interface	Task analysis; prototyping; scenarios; user characterization (functionality, style, workload)
Goldplating	Requirements scrubbing; prototyping; cost/benefit analysis; design-to-cost
Continuing stream of requirement changes	High change threshold; information hiding; incremental development (defer changes to later increments)
Shortfalls in externally furnished components	Benchmarking; inspections; reference checking; compatibility analysis
Shortfalls in externally performed tasks	Reference checking; pre-award audits; award-fee contacts; competitive design or prototyping; team building
Real-time performance shortfalls	Simulation; benchmarking; modeling; prototyping; instrumentation; tuning
Straining computer science capabilities	Technical analysis; cost-benefit analysis; prototyping; reference checking

Table 6-3. Boehm's Top-Ten Risk Identification Checklist [BOEHM91]

(4.2.2) Risk Taxonomy

The **Taxonomy-Based Risk Identification** method identifies and clarifies technical and managerial uncertainties and concerns. The software taxonomy is organized into three major classes which are further divided into elements, each of which is characterized by its attributes.

- **Product engineering**. The technical aspects of the work to be accomplished.
- **Development environment**. The methods, procedures, and tools used to produce the product.
- **Program constraints**. The contractual, organizational, and operational factors within which the software is developed, but which are generally outside of the direct control of local management. [CARR93]

The taxonomy provides a basis for organizing and studying the breadth of software development issues. As illustrated in Figure 6-4, it serves as a systematic way of eliciting and organizing risks. It provides a consistent risk management framework. Table 6-4 shows a detailed, lower-level risk taxonomy.

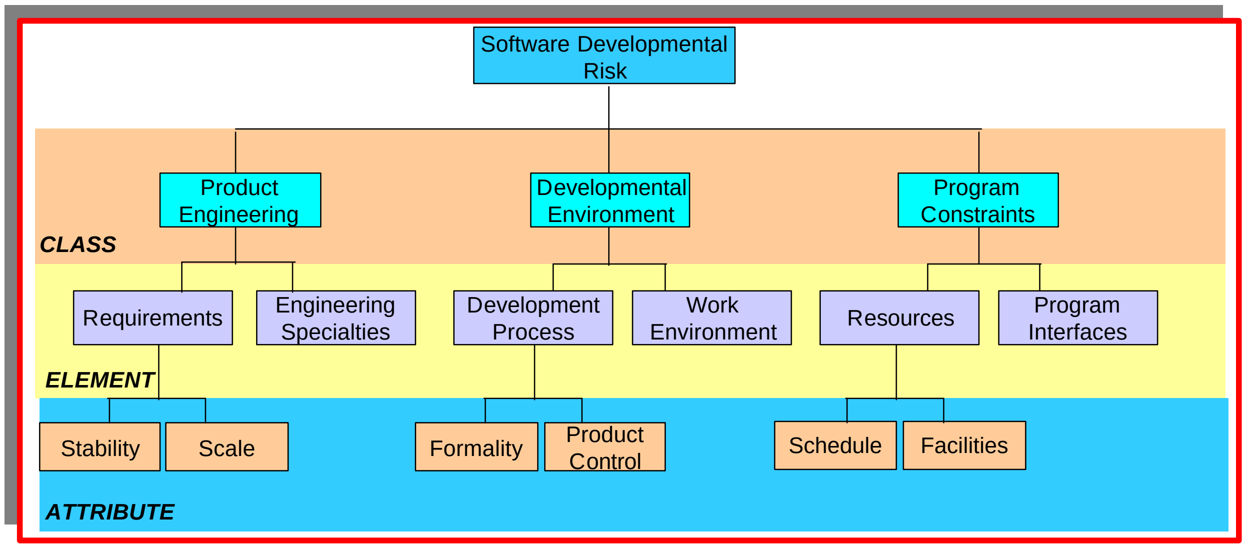


Figure 6-4. Software Risk Taxonomy Structure [CARR93]

Product Engineering	Development Environment	Program Constraints
Requirements	**Development Process**	**Resources**
• Stability	• Formality	• Schedule
• Completeness	• Suitability	• Staff
• Clarity	• Process control	• Budget
• Validity	• Familiarity	• Facilities
• Feasibility	• Product control	**Contract**
• Precedent	**Development System**	• Type of contract
• Scale	• Capability	• Restrictions
Design	• Suitability	• Dependencies
• Functionality	• Usability	**Program Interfaces**
• Difficulty	• Familiarity	• Customer
• Interfaces	• Reliability	• Associate contractors
• Performance	• System support	• Subcontractors
• Testability	• Deliverability	• Prime contractor
• Hardware constraints	**Management Process**	• Corporate management
• Non-developmental software	• Planning	• Vendors
Code and Unit Test	• Program organization	• Politics
• Feasibility	• Management experience	
• Testing	• Program interfaces	
• Coding	**Management Methods**	
• Implementation	• Monitoring	
Integration and Test	• Personnel management	
• Environment	• Quality assurance	
• Product	• Configuration management	
• System	**Work Environment**	
Engineering Specialties	• Quality attitude	
• Maintainability	• Cooperation	
• Reliability	• Communication	
• Safety	• Morale	
• Security		
• Human factors		
• Specifications		

Table 6-4. Lower Level Software Risk Taxonomy [CARR93]

6.2.3.2.1 Taxonomy-Based Questionnaire (TBQ)

A **Taxonomy-Based Questionnaire (TBQ)** is a tool specifically used for identifying risks. This tool ensures all potential risk areas are covered by asking questions at the Risk Taxonomy detailed attribute level. The example TBQ in Figure 6-5 also contains specific cues and follow-up questions that allow the person administering the questionnaire to probe for risks. This tool is effective when used along with appropriate techniques for interviewing management and technical program personnel.

A. Product Engineering
1. Requirements
a. Stability
[Are requirements changing even as the product is being produced?]
[1] Are requirements stable?
(NO) (1.a) What is the effect on the system?
- Quality
- Functionality
- Schedule
- Integration
- Design
- Testing

[2] Are the external interfaces changing?

Figure 6-5. Taxonomy-Based Questionnaire (TBQ) Example [SISTI94]

The TBQ produces better results when administered by an independent team and when the respondents are in peer group sessions. The TBQ field test process consists of four distinct activities, as illustrated in Figure 6-6.

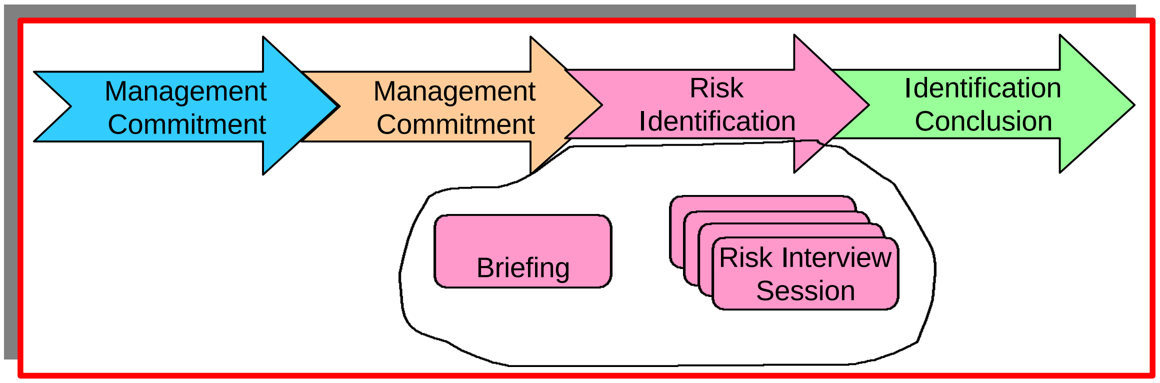


Figure 6-6. TBQ Risk Identification Process [CARR93]

More information regarding the risk taxonomy and a taxonomy-based questionnaire is available from the sources cited in the figures. Note that the foregoing checklists and taxonomy were focused largely on the developer. The program office must not forget to identify and manage risks that are solely theirs. Such risks include:

- Making sure the requirements put on contract meet the end user needs.
- Allocating sufficient schedule to properly develop the product and not surcoming to pressure to put "rubber on the ramp."
- Recognizing what a dollar will really buy.
- Selecting a supplier who has the capability to deliver what is ordered.
- Not expecting technology advancement to solve problems.
- Being prepared to deal with shifting political winds as leadership and administrations change and the perceived value of the product being developed fluctuates.
- Being prepared to deal with changes in program office staffing as personnel are reassigned.
- Assuring that final testing is done against the requirements document, not the wants/desires of the project office or end user at test time.
- Having the fortitude to tell end users and congressional staffers that changes in requirements may necessitate replanning and renegotiating the project and/or contract. This is especially true late in the project's lifecycle.

A historical review of any recent major weapon system acquisition should reaffirm the need to address the issues listed above.

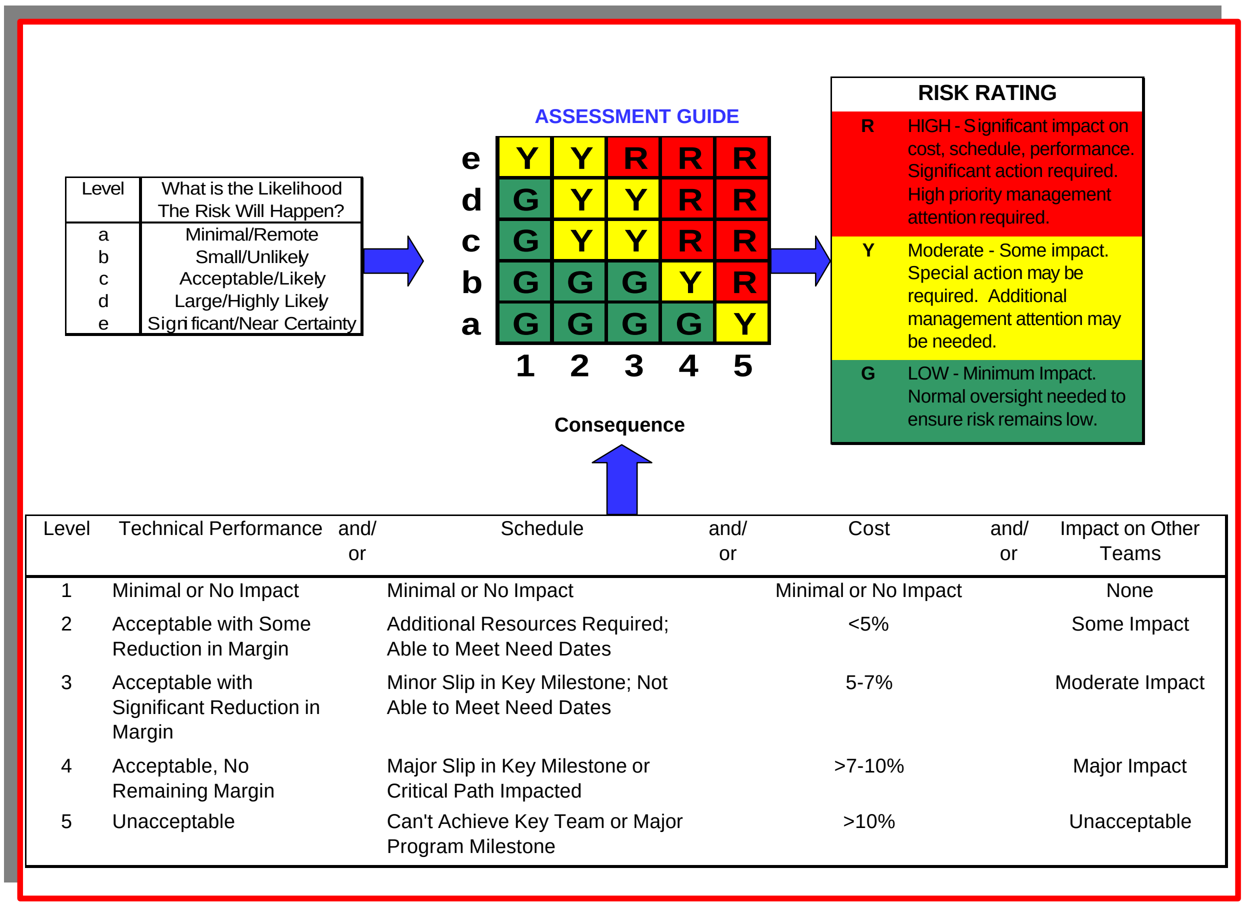
(4.3) Risk Analysis

Risk analysis is the process of examining each identified risk area to refine the description of the risk, isolating the cause, and determining the effects. Risk impact is defined in terms of its probability of occurrences, its consequences, and its relationship to other risk areas or processes.

Risks are assigned a level of magnitude (risk assessment) based on the risk rating scheme identified in the risk management plan. The scheme used for Table 6-4 below has three magnitudes: high, moderate, and low. However, a scheme with magnitudes such as critical, high, moderate, low, and no concern could be devised. The level at which a particular risk is assessed depends on the separate assessments of its severity of impact (consequences) and its probability of occurrence (likelihood).

Severity of impact is the effect of the particular risk on software performance, support, cost, and/or schedule. Each assessed risk is assigned a severity level based on the following individual factors or combinations of factors: technical performance, schedule, cost, and impact on other teams. Each factor is characterized by categories of severity of impact. For example, the technical performance factor is characterized by the following severity categories: Unacceptable, Acceptable but with no remaining margin, Acceptable with significant reduction in margin, acceptable with some reduction in margin, or minimal or no impact.

Probability of occurrence is the likelihood of the risk occurring. Each risk is assigned to a probability level: near certainty, highly likely, likely, unlikely, or remote. Table 6-5 illustrates how the overall magnitude or risk assessment is determined from the likelihood and consequence criteria.



Level	What is the Likelihood The Risk Will Happen?
a	Minimal/Remote
b	Small/Unlikely
c	Acceptable/Likely
d	Large/Highly Likely
e	Significant/Near Certainty

ASSESSMENT GUIDE

e	Y	Y	R	R	R
d	G	Y	Y	R	R
c	G	Y	Y	R	R
b	G	G	G	Y	R
a	G	G	G	G	Y
	1	2	3	4	5

Consequence

RISK RATING	
R	HIGH - Significant impact on cost, schedule, performance. Significant action required. High priority management attention required.
Y	Moderate - Some impact. Special action may be required. Additional management attention may be needed.
G	LOW - Minimum Impact. Normal oversight needed to ensure risk remains low.

Level	Technical Performance	and/ or	Schedule	and/ or	Cost	and/ or	Impact on Other Teams
1	Minimal or No Impact		Minimal or No Impact		Minimal or No Impact		None
2	Acceptable with Some Reduction in Margin		Additional Resources Required; Able to Meet Need Dates		<5%		Some Impact
3	Acceptable with Significant Reduction in Margin		Minor Slip in Key Milestone; Not Able to Meet Need Dates		5-7%		Moderate Impact
4	Acceptable, No Remaining Margin		Major Slip in Key Milestone or Critical Path Impacted		>7-10%		Major Impact
5	Unacceptable		Can't Achieve Key Team or Major Program Milestone		>10%		Unacceptable

Figure 6-7. Risk Analysis Process [DAU98]

Your risk plan should include the risk probability/severity matrix to be used in classifying the risks identified for your program/project.

(5.0) Risk Mitigation

> *"If the art of war consisted merely in not taking risk, glory would be at the mercy of very mediocre talent."* [NAPOLEON55]

Risk mitigation is the process that identifies, evaluates, selects, and implements options in order to set risk at acceptable levels given program constraints and objectives. This includes the specifics on what should be done, when it should be accomplished, who is responsible, and associated cost. The most appropriate strategy is selected from these handling options. Risk handling is an all encompassing term whereas risk mitigation and risk control are subsets of risk handling.

> *"A risk-averse culture inhibits risk management more than does the lack of a management infrastructure or a repeatable method. Such a culture generally rewards crisis management and punishes those who identify why the project might not succeed. A risk-averse culture relies on heroics to complete a project. When executive management effectively shoots the messenger who bears news of a potential risk, risk management at the project level is doomed to failure."* — Marvin J. Carr (Software Engineering Institute) [CARR97]

You must assess, rate, and decide on the possible consequences of action or inaction. You must also decide if the benefits of acting on a risk merit the expense in time and money expended. All risk handling actions (or inactions) should be documented with supporting rationale. You should also employ risk management in concert with metrics and process improvement used to measure, track, and improve the progress of your program and the development process.

> *"There is always hazard in military movements, but we must decide between the possible loss of inaction and the risk of action."* — General Robert E. Lee [LEE33]

Risk mitigation (i.e., handling) techniques include:

- **Risk avoidance**. You can *avoid the risk* of one alternative approach by choosing another with lower risk. This conscious choice avoids the potentially higher risk; however, it really results in risk reduction — not complete risk elimination. While a conscious decision to ignore (or assume) a high risk may be a creditable option, an unconscious decision to avoid risk is not. Be warned that:

> *"Software development's risky nature is easy enough to acknowledge in the abstract, but sadly, harder to acknowledge in real-world situations. Our culture has evolved such that owning up to risks is often confused with defeatism. Thus, a manager faced with a nearly impossible schedule may deliberately ignore risks to project a confident, "can-do" attitude."* — Barry W. Boehm and Tom DeMarco [BOEHM97]

- **Risk control**. You can *control risk* (the most common form of risk mitigation) by continually monitoring and correcting risky conditions. This involves the use of reviews, inspections, risk reduction milestones, development of fallback positions, and similar management techniques. Controlling risk involves developing a risk reduction plan, then tracking to that plan.
- **Risk assumption**. You can *assume risk* by making a conscious decision to accept the consequences should the event occur. Some amount of risk assumption always occurs in software acquisition programs. It is up to you to determine the appropriate level of risk that can be assumed as each situation warrants.
- **Risk transference**. You can *transfer risk* when there is an opportunity to reduce risk by sharing it. This concept is frequently used with contractors where, for instance, contract type, performance incentives (including award fees), and/or warranties are risk sharing contractual mechanisms. Although many of these techniques only share *cost risk*, risk transfer is often beneficial to the Government and the developer.
- **Pecuniary**. You can use the pecuniary means of risk aversion by setting aside a contingency fund of project resources (dollars, schedule, personnel, etc.) to be used if risks occur. This involves making a conscious decision to accept the consequences should the event occur. Some amount of risk assumption always occurs in software acquisition programs. It is up to you to determine the appropriate level of risk that can be assumed and the contingency fund required as each situation warrants.

(6.0) Risk Tracking and Control

Risk tracking is the process of systematically tracking and evaluating how your risk-handling actions compare with established risk management metrics. It also involves developing further risk-handling options, as appropriate.

- **Risk element tracking** involves the identification of your program's highest-risk issues and tracking progress towards resolving those issues through progress reports. The major benefits of risk element tracking are similar to those of cost/schedule/performance tracking, with the added benefit of identifying and maintaining a high-level of *risk consciousness*. Tracking is critical because it addresses the one risk attribute which is difficult to predict — *time*. Generalizations about risk made early in the program can, and often do, decay with time. Risk tracking keeps predictable, unpredictable, or unknown risks from becoming full-blown problems. Risk element tracking occurs after mitigation strategies and tactics have been implemented. It includes the following activities.
- **Tracking against the plan**. Assuring that the consequences of risk management decisions are the same as planned.
- **Improving plans**. Identify opportunities for refining the RAP.
- **Updating approaches**. Providing feedback for future decisions about controlling new or current risks not responding to risk mitigation or whose attributes have changed with time. [CHARETTE89]

In this section of the risk management plan, you should:

- Identify the risk tracking indicators and measures
- Identify the individuals or groups to receive the risk information
- Determine the reporting schedule

The following subsections discuss each of these activities

(6.1) Identify the risk tracking indicators and measures

Risk indicators and measures should be selected to support the goals of the risk management program. The use of the Goal, Question, Metric (GQM) [BASILI92] paradigm supports this selection. The implementing steps of the GQM paradigm are:

- Select the goals
- Identify the questions that should be asked to determine if the goals are being met
- Identify the metrics/indicators that will allow you to answer the questions

Remember that all metrics/indicators should be explicitly defined to avoid semantic issues later.

Cost/schedule/performance are the usual areas considered for measurement. This involves using techniques such as the work breakdown structure, quality indicators, activity networks, and earned-value management to determine and track program progress with respect to plans, schedules, and budgets. Cost/schedule/performance tracking is useful because potential schedule slippages, cost overruns, and performance shortfalls can be identified early, and their impact on other interdependent system elements reduced. Software development risk tracking methods include peer inspections, reviews, audits, testing, and configuration management.

(6.2) Identify the individuals or groups to receive the risk information

Identify the decision authorities who must receive the risk information. By knowing who will receive the information, you may do a better job of identifying the indicators and metrics to be collected as well as determining the required reporting schedule.

(6.3) Determine the Review and reporting schedule

The reporting should be frequently enough that the program could not go far astray between reports yet not so frequent that unnecessary work is done.

(7.0) Risk Communication

Unless those who can make decisions regarding the allocation of resources to deal with the risk are aware of the risk, preventative action cannot be taken. It is important for those who are aware of a situation to communicate the status to the decision maker. This communication must be timely. It does no good to inform a decision maker of pending calamity after the event has occurred.

Communication often takes the form of periodic program reviews. Reviews must be held frequently enough to provide necessary reaction time, or a means must be provided to communicate risk information out of the normal review cycle.

This portion of the risk management plan should address all issues of risk communication, including who, what, when, how, etc.

(8.0) Risk Feedback

Feedback on how a risk was minimized may be used to minimize similar risks. Lessons learned from risks that became problems may be used to prevent other risks from becoming problems. Additionally, feedback is necessary as risks are continuously monitored to update their status. What was a severe risk last week may be only a slight inconvenience this week while the "unlikely to occur" risk of yesterday may be an "almost certain to occur" risk today. The items to be fed back to the other risk management process activities should be identified along with the feedback schedule.

(9.0) Risk Documentation

Risk documentation involves the recording, maintaining, and reporting of risk assessments, risk-handling analysis and plans, and risk-monitoring results. It includes all plans, reports for the project manager and decision authorities, and risk reporting forms that may be internal to the program office or the contractor's management structure.

(9.1) Identify Documentation Needs

The documentation requirements for the risk management program should be detailed in this section of the plan.

(9.2) Specify documentation format

For each risk identified, you will want to create a record to track the risk. This record could be considered a risk aversion plan for each risk. The data elements for the record should be determined and documented in this section. Figure 6-7 provides some suggested record elements.

Suggested Risk Record Elements	
Element	Description
Risk Identification	A description of the perceived risk.
Risk Source	A description of the source of the risk. The source can be identified by using a risk factor checklist or risk taxonomy.
Risk Assessment	The risk assessment rating (green, yellow, red), including the probability of occurrence and potential damage, as extracted from the risk assessment matrix.
Aversion Strategy	Recommended aversion strategy with associated mitigation technique (reduction, protection, transference, pecuniary) for the risk item (also note risk items requiring no action).
Integration Strategy	An integration strategy for individual risk aversion plans (with attention to combining action plans for more than one risk item).
Resource Allocation	Assignment of resources needed to implement the risk aversion strategy (including personnel, cost, schedule, and technical considerations).
Criteria for Success	The criteria that must be met for the risk considered to be considered mitigated.
Monitoring Approach	The approach to be used to monitor progress in mitigating the risk.
Implementation Schedule	The schedule for implementing the mitigation effort, including key milestones.
Mitigation History	A record of the mitigation steps taken, the date taken, who took the steps, and their results.
Cost and Effort	The cost and effort expended in managing the risk.

Table 6-5. Risk Record Elements

(9.3) Identify the repository for the documents
The risk documents should be maintained in a central file or database. The database may be as simple as a card file or as elaborate as an online real time database. The issue is to use a database that meets the needs of your project. The Software Program Managers Network has developed Risk Radar, a database that uses Microsoft Access. It is available for free download at **http://www.spmn.com**. Other products are available commercially.

6.2.3.3 Implementing the Plan

The following paragraphs describe how to implement the Risk Management Plan by completing the remaining activities of the Risk management Process

6.2.3.3.1 Risk Assessment

Risk assessment is comprised of Risk Identification and Risk Analysis.

6.2.3.3.2 Risk Identification

The first step in implementing the plan is the risk identification step. As previously mentioned, the use of a risk factor checklist or risk taxonomy can help to minimize the chance of overlooking risk areas. The checklists and/or taxonomies from section 4.1 of the risk management plan should be used. The goal of this step is to “turn over all the rocks” to identify all potential project risks. However, recognize that all project risks may not be known at this time. That is why it is important to reinitiate this step periodically throughout the project’s lifecycle.

6.2.3.3.3 Risk Analysis

Once risks have been identified, they should be analyzed using the assessment matrix documented in section 4.2 of your risk management plan. The probability and impact should be assigned based on the best factual information available. Remember that a good dose of common sense never hurts. Because time is ever changing and the scenario may change, this assessment step must be repeated frequently during the project.

6.2.3.3.4 Risk Mitigation

Next comes the assignment of a mitigation strategy. This includes the specifics on what should be done, when it should be accomplished, who is responsible, and associated cost. The mitigation strategies to be considered should be those from section 5 of your risk management plan.

All risks should be assigned a mitigation technique, including the technique of “no action.” Sometimes a combination of techniques is more effective than a single technique. Those risks with an analysis rating of red should have a detailed mitigation plan developed and implemented. Risks with an analysis rating of yellow should have basic mitigation plans developed and implemented. Remember that resources can be saved by implementing mitigation strategies that address multiple risks together.

Some may refer to the risk mitigation plan as a contingency plan. A contingency plan is really a combination of several of the risk mitigation techniques discussed previously. The idea is to have an alternative solution available if the originally planned solution does not succeed.

Napoleon’s Military Maxim Number 8 states:

> *“A general-in-chief should ask himself frequently in the day, ‘What should I do if the enemy’s army appeared now in my front, or on my right, or on my left?’ If he has any difficulty in answering these questions he is ill posted and should seek to remedy it.”* — Napoleon Bonaparte, I [NAPOLEON31]

Roetzheim provides an example of contingency planning when a program manager to develop a large software-intensive system for the Navy. The system was to graphically plot specific target positions by monitoring tactical Navy data links. He was told the Government would supply the software interface (to be designed by government personnel) to convert the existing electronic sensor signals into a compatible format with the software system under development.

After interviewing the government design team, Roetzheim was convinced they did not have the necessary experience in the specialized Navy system to perform the task successfully. He concluded that the probability of failure for this critical item was fairly high. The situation was further compounded by the fact that his software development program was on a tight schedule. Final system delivery was an at-sea demonstration during a major fleet exercise. In short, the consequences of failure for his program were extreme. One month before the full-scale demonstration, his worst fears were confirmed. The government-supplied interface failed dismally during the first live test run. Luckily, his contingency planning included the following actions.

- He had company engineers rough out an initial interface design in the event in-house development was required;
- Based on the initial design, the company archives were searched and all required technical data and specialized components were located and on-hand;
- Company personnel capable of performing the task were identified and their availability was confirmed if they were needed for an emergency job;
- Overtime for the engineering department was approved by management; and
- A development plan, including all cost estimates and a statement of work, were prepared and all the information needed to write a delivery order was ready.

The afternoon of the live test run failure, Roetzheim convinced the Navy contracting officer to allow him to begin work on his interface in parallel with the government staff. As feared, the government-furnished interface did not work. The at-sea demonstration was successfully completed using the in-house developed interface. If he had not identified this high-risk item and performed adequate contingency planning, his software program would have been a high-profile, embarrassing fiasco — damaging his reputation and possible future work for his company. [ROETZHEIM88]

Fairley explains that contingency planning must include justification for any added cost incurred in preparing a the plan, monitoring the situation, and implementing action plans, if necessary. If the cost is justified (as in Roetzheim's case where the cost of failure was extreme), immediate plan preparation and implementation might be necessary. This brings up the issue of *adequate lead-time*. When should Rotzheim have started development of his own interface? The answer comes from continuous analysis and tracking of the probability of failure and its consequences. As the probability becomes greater, the urgency to build his interface becomes greater. With proper contingency planning, drop-dead dates are established, funds and personnel are set aside, while staying within preplanned delivery schedules. [FAIRLEY94]

A risk record (see your risk management plan, section 9.2) should be completed for each risk and the record should be placed the document repository or database as described in section 9.3 of your risk management plan.

6.2.3.3.5 Risk Tracking and Control

Risks should be tracked and controlled per section 6 of the risk management plan. The measurements and indicators identified in section 6.1 of the risk management plan should be used. Note that some programs have established thresholds to determine when additional mitigation efforts should be implemented. Recognize that it is not just the threshold, but the rate at which the threshold is being reached, that should determine when additional effort is required. For example, when driving down the highway, the threshold is the lane markers. You may determine that you are well within the lane and that you will not cross the lane marker for another quarter mile. You have plenty of time to react. A slight correction at this time may extend the distance before you cross the lane marker to a half mile. You have no great concern. However, if you hit a stretch of icy road and your car begins to spin, you may still be within the lane markers but may be expected to cross the lane marker within fifty feet. Immediate attention and mitigation effort is needed NOW. You cannot afford to wait until the lane marker, or threshold, is exceeded.

6.2.3.3.6 Risk Communication

Risk review and reporting should occur according to the schedule documented in section 6.3 of the risk management plan. Those who are involved in the reviews or who receive the reports are identified in section 7 of the risk management plan. All risks in the red range, along with the near-red yellow risks, should be discussed during each review meeting. This means that there should be no arbitrary limits placed on the number of risks to be discussed such as a "top ten" list. Why? If a risk truly warrants a red evaluation during the risk analysis phase or later re-evaluation, it is important enough for management to be concerned. The current status of each risk, the ongoing mitigation effort, and the success of the mitigation effort should be included in the discussion.

6.2.3.3.7 Risk Feedback

Lessons learned, either positive or negative, should be recorded and used in future risk evaluations.

6.2.3.3.8 Risk Documentation

Risk documentation should be updated as the risk assessment changes with time and as decisions are made during the risk reviews.

6.2.4 Crisis Management Plans

A crisis is an overall show-stopper! All program effort and resources must be focused on resolving a crisis situation. Once in crisis, you must gather your forces, go on the offensive, and *attack!* If you do not attack this type of risk, it will attack you — and win!

A crisis occurs when your Contingency Plan fails to resolve an unforeseen event. If you do not act quickly to manage a major unforeseen negative event, you may as well resign and go home. If Roetzheim was unable to allocate enough engineers or hours to develop the interface, his Contingency Plan would have failed.

If Roetzheim had waited for the government-supplied interface to fail before building his own interface, he would have been out of time, over budget, and still not have the interface needed to complete the at-sea demonstration successfully. Before a crisis materializes, you may be able to define some elements of crisis management, such as the responsible parties and a drop-dead date. However, you may be hard pressed to develop the exact details of a Contingency Plan until a crisis occurs. If you are in such a situation, Fairley recommends the following.

- Announce and publicize the problem;
- Assign responsibilities and authorities;
- Update status frequently;
- Relax resource constraints (fly in experts, bring on emergency personnel, provide meals and sleeping facilities to keep them on site until the crisis is resolved, etc.);
- Have program personnel operate in burnout mode;
- Establish a drop-dead date; and
- Clear out nonessential personnel. [FAIRLEY94]

6.2.5 Crisis Recovery Procedure

Once recovered, you must examine what went wrong, evaluate how your budget and schedule have been affected, and reward key crisis management personnel. During crisis recovery, you should do the following:

- Conduct a crisis postmortem;
- Fix any systematic problems that caused the crisis;
- Document lessons-learned;
- Recalculate program cost and schedule;
- Rebaseline; and
- Update your schedule and cost estimates to reflect the new baseline. [FAIRLEY94]

6.3 Software Acquisition Risk Management

DoD 5000.2-R states that Program Managers must establish a risk management program for each acquisition to identify and control performance, cost, and schedule risks. Risk management programs must:

- Identify and track risks and risk drivers,
- Define risk aversion plans, and
- Provide for continuous risk assessments during each acquisition phase to determine how risks have changed.

The following paragraphs identify some common risk factors that are common to most large-scale software acquisitions. Be aware, these sources of risk are provided as a starting point. You should also identify those risks that are unique to your specific program. The ability to identify program-specific risks is predicated on your knowledge of your solution domain and your product. The greater knowledge you have, the greater your ability to identify risks. The maturity of your team's technical and managerial processes is also a major factor. While some of these areas may overlap, they all must be considered and managed.

Regardless of the selected acquisition strategy, the following should be considered high risk acquisitions.

- **Software-intensive systems**. Systems with a large software component are often delivered late, over-budget and do not satisfy user expectations.
- **Rapidly changing hardware/software technologies**. Traditional acquisition models often result in the delivery of systems with obsolescent technology, due to the high speed of technology advancement. Such systems often require subsequent upgrading at substantial cost.

- **Human-in-the-loop (HIL)**. Systems where humans are an integral component. Where the performance of hardware/software performance may be predictable, the effects user interaction with the technology is more difficult to predict. This makes system requirements difficult to define. In addition, system shortcomings are often not revealed until operational testing. Or worst case, they are not discovered until extended operational use. This leads to costly requests for post-deployment modifications. Similarly, users often see opportunities for capabilities not originally envisioned, leading to requests for enhancements or upgrades.
- **Systems with many diverse users**. Variations in the experience and competence of users makes defining user requirements difficult, particularly in the area of human-computer interface (HCI) design. This problem is exacerbated when there is a large number of users.
- **Unprecedented systems**. Users have difficulty accurately identifying requirements for systems that will satisfy unprecedented requirements or for which they have little or no experience.
- **Systems providing a quickly needed, limited capability**. Operational demands dictate that a limited capability is required early. [HENDERSON97]

ATTENTION! Risk management must be an on-going process. It is critically important to combine risk management with acquisition planning discussed in Chapter 7, *Software Acquisition Planning*. Be sure to read and implement Chapters 6, *Risk Management*, and 7 together.

Your Acquisition Strategy should include a Risk Management Plan that identifies key acquisition risk drivers in critical risk components and processes (which may be the sum of a series of smaller risks). The plan discusses how risks will be managed, with particular attention to interrelationships among lower level events that can quantifiably impact program success. Table 6-6 summarizes the critical risk areas for which the probability and consequences of acquisition failure must to be determined for all major DoD acquisition programs.

Risk Area	Significant Acquisition Program Risks
Threat	● Uncertainty in threat accuracy and stability ● Sensitivity of design and technology to threat ● Vulnerability of system to threat countermeasures ● Vulnerability of program to intelligence penetration
Requirements	● Operational requirements not properly established or vaguely stated for program phase ● Requirements are not stable ● Required operating environment not described ● Requirements do not address logistics and suitability ● Requirements are too constrictive-identify specific solutions that force high cost
Design	● Design implications not sufficiently considered in concept exploration ● System will not satisfy user requirements ● Mismatch of user manpower or skill profiles with system design solution or human-machine interface problems ● Increased user skills or more training requirements identified late in the acquisition process ● Design not cost effective ● Design relies on immature technologies to achieve performance objectives
Test & Evaluation (T&E)	● Test planning not initiated early in program (Phase 0) ● Testing does not address the ultimate operating environment ● Test procedures do not address all major performance and suitability specifications ● Test facilities not available to accomplish specific tests, especially system-level tests ● Insufficient time to test thoroughly
Modeling & Simulation	● Same risks as identified for T&E ● M&S are not verified, validated, or accredited for the intended purpose
Technology	● Program depends on unproven technology for success or there are no alternatives ● Program success depends on achieving advances in state-of-the-art technology ● Potential advances in technology will result in less than optimal cost-effective system or make system components obsolete ● Technology has not been demonstrated in required operating environment ● Technology relies on complex hardware, software, or integration design ● Program lacks proper tools and modeling and simulation capability to assess alternatives
Logistics	Inadequate supportability late in development or after fielding, resulting in need for engineering changes, increased costs, and/or schedule delays Life-cycle costs not accurate because of poor logistics supportability analyses (LSA) LSA results not included in cost-performance tradeoffs Design trade studies do not include supportability considerations
Development	Development implications not considered during concept exploration Development not sufficiently considered during design Inadequate planning for long lead items and vendor support Development processes not proven Prime contractors do not have adequate plans for managing subcontractors Sufficient development tools not readily available for cost-effective production Contract offers no incentive to upgrade tools, improve processes, or reduce costs
Concurrency	Immature or unproven technologies will not be adequately developed prior to system production Development funding will be available too early-before the development effort has sufficiently matured Concurrency established without clear understanding of risks
Developer Capability	● Developer has limited experience in specific type of development ● Contractor has poor track record relative to costs and schedule ● Contractor experiences loss of key personnel ● Prime contractor relies excessively on subcontractors for major development efforts ● Contractor will require significant capitalization to meet program requirements

Table 6-6. Significant Acquisition Risks to Address in the Acquisition Strategy

Haimes and Chittister explain that software acquisition program managers must be able to verify offeror's cost and schedule estimates to make informed decisions about the risk of program cost and schedule overruns. You should be concerned with the non-technical acquisition risks addressed by the following questions.

Based on the offeror's level of experience, the assumptions made, the estimating models selected, and how model parameters are defined, do software developers with minimal experience overestimate or underestimate the complexity of the proposed software development task? What are the sources of software development cost estimation risk? How can this risk be quantified? [HAIMES95]

6.3.1 Risk-Based Source Selection

Andy Mills [U.S. Army Communications and Electronics Command (CECOM) Software Engineering Directorate] defined a risk-based source selection approach where offerors' technical proposals are evaluated and awarded based on how effectively they address software risk management. In the RFP, he suggests that offerors be evaluated on how risk will managed throughout the contract. During the evaluation process, the Government must implement its own risk management program. By applying a standardized risk management approach to source selection, a thorough evaluation can be achieved that increases the effectiveness of acquisition streamlining.

6.3.1.1 Risk-Based RFP

The success of software-intensive system acquisitions depends on the establishment of a planned acquisition path that manages risks. By requiring a risk-based approach, offerors' proposals should state how they would plan and schedule software activities based upon realistic assessments of technical challenges and risks. Their Risk Management Plan should describe how they plan to attack software risks through an appropriate choice of software architectures, reuse strategies, requirements management processes, metrics, development models, tools, and technologies. In fact, the offeror's proposal is an initial Software Development Plan (SDP). Subsequent adjustments of the approach often become necessary, as program risks shift during development. However, a generic approach that does not plan for program and product risks will result in having to resolve otherwise foreseeable problems if resources are already spent or committed elsewhere. [MILLS95]

To understand what is meant by requiring a proposal based on risk management, assume that every software development activity is traceable to the mitigation of one risk or another. The RFP should state that the offeror's approach be organized around identified software development risks and how they will exploit risk mitigation opportunities throughout contract performance. Table 6-7 provides a typical list of proposal requirements. These areas represent possible areas for mitigating software development risks. From industry's perspective, the proposal tells the Government why the proposed approach is the best possible. For each element of the proposal, the offeror should explain how the proposed activity contributes to risk reduction. [MILLS95]

Typical Software Development Proposal Requirements
Requirements difficulty
1. Software architecture
2. Domain reuse
3. System capacity and resources
4. Software engineering environment
5. Software tools
6. Proprietary versus Government data rights
7. Government furnished items/software
8. Commercial-off-the-shelf software
9. Interface requirements, integration, & support
10. Process improvement
11. Testing
12. Corrective action process
13. Configuration management
14. Software quality evaluation
15. Preparing for software transition
16. Risk management implementation
17. Software element tracking
18. Technical reviews
19. Subcontractor management
20. Program organization and resources
21. Personnel qualifications/experience/availability
22. Software work breakdown structure (WBS)
23. Engineering procedures
24. Planning and replanning: • Identifying software activities • Estimating activities • Scheduling software activities • Activity network

Table 6-7. Areas Where Risk Mitigation Can Be Addressed in the RFP [MILLS95]

6.3.1.2 Risk-Based Proposal Evaluation

Proposal evaluation has traditionally depended on the opinions of highly qualified technical teams participating in the source evaluation and selection process. Using a software risk evaluation methodology extends the capabilities of evaluation personnel by facilitating risk identification and by focusing attention on the feasibility and merits of the offeror's approach.

Mills explains that when offerors approach the proposed software development from a risk management perspective, proposals can be readily evaluated for their risk mitigation strengths (and/or weaknesses). Although technical proposals are currently systematically evaluated, source selection teams should implement a risk management approach that identifies and assesses risk, such as the SEI's Software Development Risk Taxonomy and the associated Taxonomy-Based Questionnaire (both discussed above). The TBQ addresses source selection risk and poses critical questions within each detailed taxonomy attribute. These are intertwined within the risk reduction and management activities for each taxonomy topic. This permits a detailed evaluation of the feasibility of a proposed approach without prescribing required activities or practices.

Source selection must be based on balancing performance, schedule, and cost objectives by selecting the contractor team that provides the best value to the user within acceptable risk limits. Therefore, the source selection team must evaluate each offeror's capability for meeting product and process technical, schedule, and cost requirements while addressing and controlling inherent software development risks. The evaluation team should discriminate among offerors based on the following:

- Product and process approach and associated risks determined by comparison with the best practices baseline;
- Ability to perform with a focus on the critical risk elements inherent in the program;
- Adherence to requirements associated with any mandatory legal items; and
- Past performance on software development efforts of similar in size, complexity, and software domain to the proposed program being evaluated.

The process of choosing among offerors is significantly enhanced if the evaluation team includes risk management as a "source selection discriminator" to be used in making the selection decision. Risk management then becomes an important factor in the Source Selection Authority determination of who can provide the most risk managed product.

McConnell explains that experienced software developers have also accumulated years of hard-won experience. Software development programs often contain uncharted paths and dangerous territory. He proposes the following list of Ten Essentials for software successful software development programs. These should all be addressed in the RFP and offeror's proposals should be evaluated on how the approach these essentials:

Software's Ten Essentials	
1	A product specification
2	A detailed user interface prototype
3	A realistic schedule
4	Explicit priorities
5	Active risk management
6	A quality assurance plan
7	Detailed activity lists
8	Software configuration management
9	Software architecture
10	An integration plan

Table 6-8. Ten Essentials for a Successful Software Development Program [MCCONNELL97]

6.3.2 B-1B Computer Upgrade Program Software Acquisition Risk Management

In June 1992, the Air Force identified new conventional roles for the nuclear deterrent bombers, the B-1B and the B-2A. Since then, the B-1B program has focused a series of systematic upgrades to develop new weapons delivery software. Judicious risk management has been practiced in all phases of the Conventional Munitions Upgrade Program (CMUP) upgrade program.

6.3.2.1 Identified B-1B Risks

Many risk management lessons have been learned on the B-1B upgrade program. One risk, true for any acquisition program, is *funding*. Ensuring that available funding is not squandered on mistakes or misdirections requires a *well-defined set of requirements* and *active management involvement*. To help reduce program cost risk, the B-1B has maximized the use of COTS technology

Another risk item was the Perry Memo, which requires the use of best commercial practices wherever possible and to use Mil-Specs only when no suitable commercial standard exists. The B-1B upgrade was one of the first programs to implement this policy. In some cases, equivalent commercial standards and practices were difficult to identify. The issue was compounded by the fact that many Defense contractors did not compete commercially (at least not in the same division); thus they were not familiar with current commercial practices.

The CMUP also required compliance with the public law mandating the use of the Ada programming language. If modifications to existing and/or new code exceed 33% the size of the CSCI, existing JOVIAL code had to be converted to Ada. The new CPU could not support the required code conversion to Ada. The code conversion was a good idea from the supportability standpoint, but it required regression testing and recertification, especially for the terrain-following and nuclear software. Other lessons learned include the following.

- You can never have too much spare memory and throughput;
- Never rely on contractor claims that firmware items will not require system life cycle adaptation; and
- Conducting a *fly-fix-fly* flight test program is very inefficient. [STORMONT95]

6.3.2.2 Contractor Risk Management Teams

The prime contractor assumed the task of administering the Risk Management Program. Integration control teams (government/contractor) identified and managed risk areas, as illustrated in Figure 6-8. Risks were also identified and tracked at the product level by product teams and at the systems level by the Systems Engineering Team. [STORMONT95]

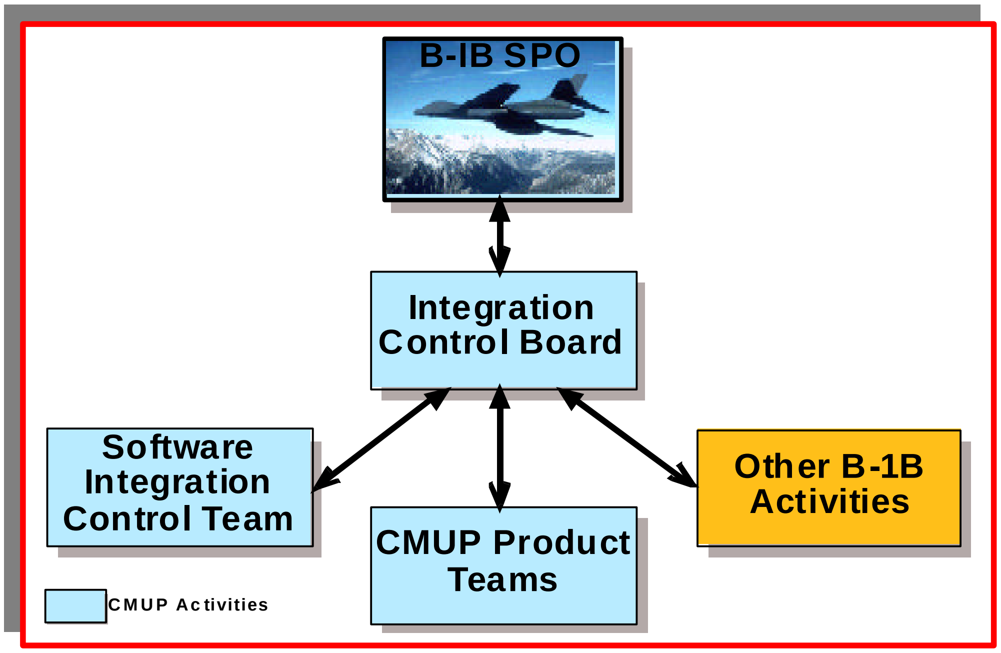


Figure 6-8. CMUP Risk Management Teams

6.3.2.3 Integrated Risk Management Process

The Integrated Risk Management Process (IRMP) was an Aeronautical Systems Center (ASC) initiative. The B-1B CMUP was the first program to successfully incorporate the IRMP as part of its contracting process. Experts from the engineering and financial communities performed technical content, budget, and program schedule evaluations. The SPO worked with the evaluators to mitigate risk areas identified in the IRMP. This integrated process provided a vehicle for honest evaluations and program risk analyses without the *finger-pointing* that frequently occurs when outside agencies evaluate a SPO's efforts. The IRMP assessment identified several risk areas requiring SPO attention, which were addressed before CMUP contract award. [STORMONT95]

6.3.2.4 Chief Engineers Watchlist

The B-1B program put a new spin on the old *watchlist* technique. Risk item lists were maintained by product and systems engineering teams. However, the Chief Engineers Watchlist allowed contractor chief engineers and the SPO to identify and track important risk items not on the other watchlists. Technical personnel at the management level often identify risks not apparent at the working level. This list ensures these items are not overlooked by product teams or systems engineers. [STORMONT95]

Figure 6-9. B-1B Conventional Munitions Upgrade Software in Action [USAF Photo]

6.4 Software Development Risk Management

EIA/IEEE J-STD-016 states:

> *"The developer shall perform risk management throughout the software development process. The developer shall:*
> - *Identify, analyze, and prioritize the areas of the software development program that involve potential technical, cost, or schedule risks;*
> - *Develop strategies for managing those risks;*
> - *Record the risks and strategies in the software development plan; and*
> - *Implement the strategies in accordance with the plan."*

While you can never totally remove software risk, there are many techniques that can be used to mitigate it. These techniques should be used in the structure of a software risk management process. While many methods exist, the next section, *Formal Risk Management Methods*, presents several structured, well-proven risk methods from which to choose. Be advised, a formal risk management method must be tailored to your specific risk environment, as illustrated by the examples listed in the "*Applied Risk Management*" section below.

ATTENTION! Effective risk management, measurement, and metrics go hand-in-hand. If you are implementing a software development risk management program, be sure to read Chapter 13, *Software Estimation, Measurement, & Metrics.*

6.4.1 Software Development Risk Factors

Software risk factors that impact a product's performance, cost, and schedule can be further segmented into five risk areas. However, any given risk may have an impact in more than one area. The five risk areas are:

- **Technical risk**. On implementation, the software system does not perform as originally intended, or it is so user-hostile it is under-used or rejected by users.
- **Supportability risk**. The software is so poorly constructed or complex, it is too costly or impossible to maintain, upgrade, or maintain.
- **Programmatic risk**. Cost, schedule, or size estimates are so unrealistic the program is under funded.
- **Cost risk**. The cost of development exceeds any benefits the system may offer during its useful life.
- **Schedule risk**. The system is not delivered on time, making other system elements late and/or the user opts to use alternative solutions.

Charrette explains that subtle environmental factors are often overlooked when identifying sources of risk. These environmental factors include the following.

- **Software developments are very complex**. The software problem domain has numerous elements with extremely complicated interrelationships.
- **Problem element relationships are multidimensional**. The changes in elements are not governed by the laws of proportionality. For example, adding more people to a program that is behind schedule, in many instances may make it even later.
- **Software problem elements are unstable and changeable**. Although cost and schedule may be fixed, actual costs in labor and time to complete are difficult to project.
- **The development process is dynamic**. Conditions ceaselessly change. Thus, program equilibrium is seldom achieved. The environment is never static (e.g., hardware malfunctions, personnel quit, contractors do not deliver, etc.).
- **People are an essential software development element and a major source of risk**. Economic or technical problems are relatively easy to deal with. The higher-level complications, multidimensional ambiguities, and changing environment caused by conflicting human requirements, interaction, and desires cause problems. Software development is full of problems because it is a very *human endeavor*. [CHARETTE89]

Additionally, there are other interrelated factors that contribute to software risk. These factors include:

- **Communications.** Communicating risk is one of the most difficult, yet important, practices you must establish in your program. People do not naturally want to talk about potential problems. Rather than confronting potential problems while they are still in the risk stage, we wind up having to deal with them when they become full-blown problems. Then there is a lot of communication! Effective risk planning only occurs when people are willing to talk about risks in a non-threatening, constructive environment.

- **Software size.** Size growth affects the accuracy and efficacy of estimates. Interdependence among software elements increases exponentially as size increases. With extremely large software systems, handling complexity through decomposition becomes increasingly difficult because even decomposed elements may be unmanageable.
- **Software architecture.** Architectural structure is the ease with which functions can be modularized and the hierarchical nature of the information to be processed. It also includes development team structure, its relationship with the user and to one another, and the ease with which the human structure can develop the software architecture. [PRESSMAN93]

Table 6-9 lists the software acquisition risk issues Conrow and Shisnido have classified for large-scale DoD acquisition programs.

Risk Grouping	Software Risk Issues
Program Level	• Excessive, immature, unrealistic, or unstable requirements • Lack of user involvement • Underestimation of program complexity or dynamic nature
Program Attributes	• Performance shortfalls (includes defects and quality) • Unrealistic cost or schedule (estimates and/or allocated amounts)
Management	• Ineffective program management (multiple levels possible)
Engineering	• Ineffective integration, assembly and test, quality control, specialty engineering, or systems engineering (multiple levels possible) • Unanticipated difficulties associated with the user interface
Work Environment	• Immature or untried design, process, or technologies selected • Inadequate work plans or configuration control • Inappropriate methods or tool selection or inaccurate metrics • Poor training
Other	• Inadequate or excessive documentation or review process • Legal or contractual issues (such as litigation, malpractice, ownership) • Obsolescence (includes excessive schedule length) • Unanticipated difficulties with subcontracted items • Unanticipated maintenance and/or support costs

Table 6-9. Summary of Key Risk Issues on Large-Scale Defense Software-Intensive Development Programs [CONROW97]

6.4.1.1 Common Thread Risk Factors

The Software Program Manager’s Network (SPMN) has identified common threads among troubled software programs, which include the following.

- **Management**. Management is inconsistent, inappropriately applied or not applied at all. In many cases, it is reactionary. Management reacts to, rather than plans for, issues.
- **Predictable risks ignored**. When a problem arises, it is identified by program personnel; but they often say, “*Hey, it’s too much trouble. I can’t deal with it.*” They ignore it, press on, and are blind-sided by the impact.
- **Disciplines not uniformly implemented**. Configuration management, product assurance, and technical disciplines are not uniformly implemented. Organizations throw away quality standards to “*buy*” more schedule time, performance, or save on cost. The program moves along in the short-term, but creates a rolling wave of disaster in the long-term.

- **Poor training**. In many cases, the reason managers do not perform a specific task is they do not know how to do it. For example, some managers do not understand costing or scheduling, how the technical job should be performed, or how to plan. Access to, or resources for, manager training are not available.
- **Fallacy of easy solutions**. Software programs often get in trouble when generic solutions are applied to specific problems. Methods designed for non real-time work are used in real-time programs. Unproven techniques with no tool support, standards, or an experience base from which to proceed are used on high-risk programs. Programs in trouble fail to scale the work to available resources.
- **Inadequate work plans**. Critical constraints and work plans must include schedules, budgets, work allocation, and limited time-sensitive resources. Programs in trouble do not have a clue where they stand. Their schedules are inadequate or schedules not enforced.
- **Schedule reality**. The schedule plan must be realistic. If the schedule slips, the impact on delivery must be assessed. If there is a schedule slip, programs in trouble do not realistically consider the effect on the end-date. They take short cuts and came up with success-oriented schedules to avoid announcing an end-date slip.
- **Delivery focus**. Troubled programs focus on schedule and process, not the delivery. Successful programs focus on the incremental completion of an event. The relationship among all activities should be on the completion of an event, which transfers to the next completion, and the next.
- **Constraints**. Successful programs use reasonable measurements to analyze and determine program status, assess product quality and process effectiveness, and to program the potential for success. Programs in trouble abuse measurement results, which are used to justify unreasonable positions. Bad measures are ignored and good ones are used to inflate to overshadow the bad.
- **Government responsibility**. The government Program Manager (PM) should not just sit back and oversee. The PM has to provide the hierarchy of documentation and provide standards for the software development phase. In troubled programs, the Government has a "*hands-off*" approach.
- **Methods and tool selection**. Troubled programs select tools that are inappropriate for the job. The program staff has too little or no experience in the methods used, which are not integrated and run as stovepipes. The process is also not integrated through configuration management. No effective information flow within the program is established. [EVANS94]

6.4.2 Formal Software Risk Management Methods

"Software process describes that which is common from project to project. Risk management describes what is different about your project from all others. What is your unique set of circumstances and issues for this particular effort? How will you handle them and how will that affect your plans? Given that risk management addresses the particular and not the general, beware of anyone who tries to turn the practice into a wholly standard process. Anyone who offers you the complete formula for risk management has never even heard of your project, so how can he or she be certain of identifying your risks before they've even come into being? Effective risk identification needs the participation of all informed parties and a brainstorming, no-holds-barred atmosphere." — Tim Lister [LISTER97]

Formal risk management refers to a structured process whereby risks are systematically identified, analyzed, and controlled. A structured risk management process must be implemented early, continuously, and rigorously. It provides a disciplined environment for decision making and for the efficient use of program resources. Through a disciplined process, obscure and lower-level risks that collectively could pose a major risk can be identified.

A proactive approach is the most effective way to control risks. The methods employed must be disciplined, systematic, repeatable, and based on proven practices. They must facilitate communication among all stakeholders — at all levels. A disciplined risk management approach will help you to obtain valuable program insight, to make decisions, and to take actions that may be critical to program success.

Formal risk management methods provide the necessary information to focus on priority risks and their mitigation. They define distinct procedures for performing risk management functions and provide integrated tools and techniques to ensure standardized application. Formal methods gain their robustness through practical use, testing, and continuous field validation. A formal risk management method should produce similar, consistent results no matter who applies them. There are many risk management methods. Keep in mind that whichever method select, it must be tailored to specific program needs. Examples of these methods are discussed here.

- SEI Risk Management Methods
- Software Risk Evaluation (SRE) Method
- Team Risk Management (TRM) Method
- Continuous Risk Management (CRM) Method
- Boehm’s Software Risk Management Method
- Software Program Managers Network’s (SPMN’s) Risk Management Method

6.4.2.1 Software Risk Evaluation Method

The Software Engineering Institute (SEI) **Software Risk Evaluation (SRE)** method focuses on the contractor/government relationship. It is a formal approach for identifying, analyzing, communicating, and mitigating software-intensive acquisition technical risks. This method views risks from a system level. It identifies representative risk areas in a system environment as software, hardware, technology, cost and schedule, and people, as illustrated in Figure 6-10. The risks in these areas have complex interactions and interdependencies with each other.

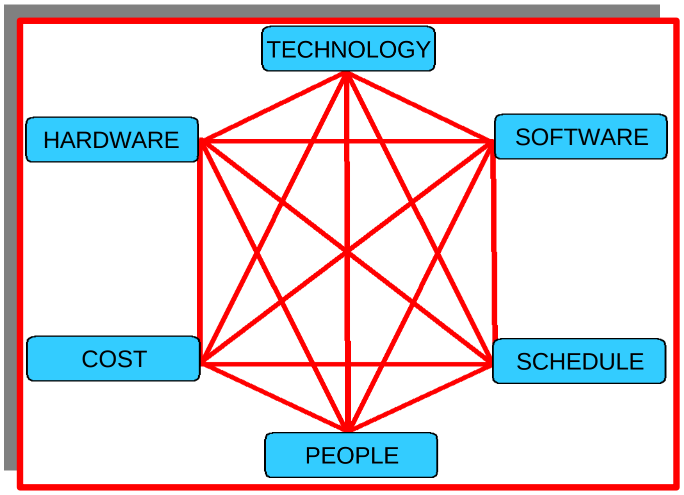


Figure 6-10. SEI-Identified Software Program Risk Areas [SISTI94]

As illustrated in Figure 6-11, the Program Manager (government) directs an independent SRE team to perform the risk evaluation. The team then executes SRE functions for the contractor's target software development task. The outcome is a set of findings processed to provide value-added information (results) to the Government. The SRE can also be used as a business tool by contractors to manage software program risks.

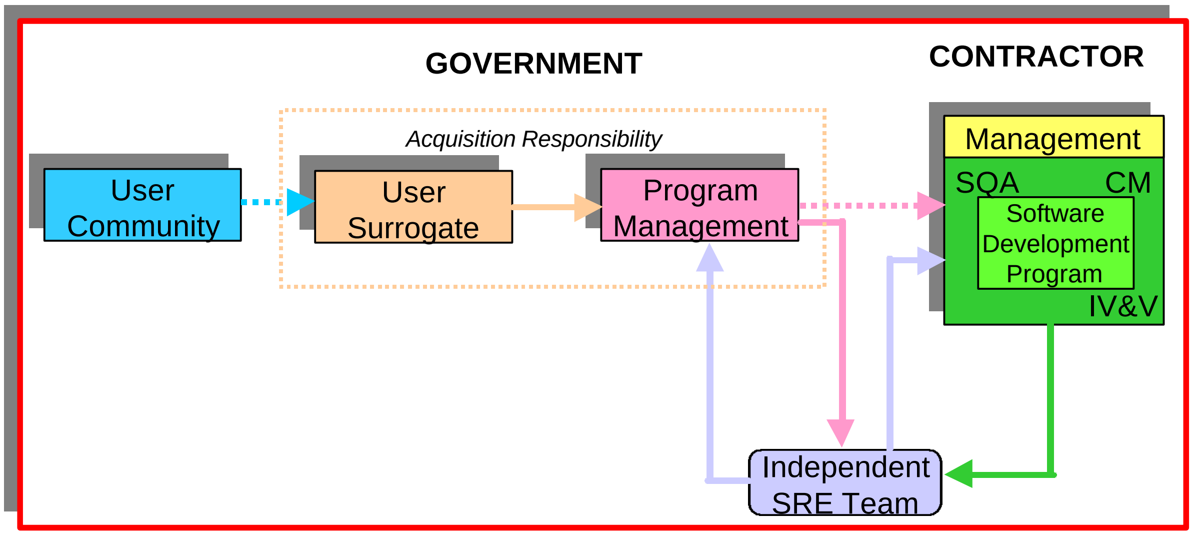


Figure 6-11. SRE Government/Industry Risk Management Method

Risk are assessed at one of three levels of magnitude: high, medium, or low. The level at which a particular risk is assessed depends on the separate assessments of its severity of impact and its probability of occurrence. *Severity of impact* is the effect of the particular risk on the target program or task and its impact on software performance, support, cost, and/or schedule. Each assessed risk is assigned a severity level: catastrophic/critical, marginal, or negligible. *Probability of occurrence* is the likelihood of the risk occurring. Each risk is assigned to a probability level: very likely, probable, or improbable.

Magnitude	Guidelines
Critical	• High likelihood of severely impacting one or more factore, i.e., cost & schedule, performance, or supportability.
High	• High likelihood of moderately impacting one or more factors.
Medium	• Medium likelihood of moderately impacting one or more factors.
Low	• Low likelihood of moderately impacting one or more factors.

Table 6-10. Level of Risk Magnitude and Guidelines [SISTI94]

6.4.2.2 Team Risk Management (TRM)

The SEI **Team Risk Management (TRM)** views the joint government/industry team approach as a constant process throughout the acquisition life cycle. Effective communication about risk is addressed from a software perspective. The objective o TRM is to create a non-threatening atmosphere for risk control and mitigation. With this method, plans are put in place to control risks while they are still manageable.

ATTENTION: Although it is desirable to have provisions for TRM as part of the contract, the SEI has concentrated their work on building the team environment after contract award and the team is in place.

6.4.2.2.1 Team Risk Management Principles & Benefits

Under the TRM method, the Government and contractor perform risk management together. The seven principles of TRM are illustrated in Table 6-11.

Risk Principle	Effective Risk Management Requirement
Shared Product Vision	Sharing product vision based upon common purpose, shared ownership, and collective commitment Focusing on results
Teamwork	Working cooperatively to achieve a common goal Pooling talent, skills, and knowledge
Global Perspective	Viewing software development in the context of a larger systems-level definition, design, and development Recognizing the potential value risk mitigation and the potential impact of adverse effects
Forward-Looking View	Thinking to tomorrow, identifying uncertainties, anticipating potential outcomes Managing program resources and activities while anticipating uncertainties
Open Communication	Encouraging free-flow of information at and between all program levels Enabling formal, informal, and impromptu communication Using consensus-building processes that value the individual (bringing unique knowledge and insight to identifying and managing risk)
Integrated management	Making risk management an integral and vital part of program management Institutionalizing risk management methods and tools in the program infrastructure and culture
Continuous process	Sustaining constant vigilance Identifying and managing risks routinely throughout all program life cycle phases

Table 6-11. Team Risk Management Principles [HIGUERA94]

Table 6-12 highlights common benefits found on programs implementing TRM. [HIGUERA94]

Benefit	Description
Improved Communications	The government and industry draw on each other's resources to mitigate risks, which enables rapid response to developing risks or problems.
Multiple Perspectives	Bringing the government and industry together opens doors to strategies they can perform together, but can not do alone.
Broader Base of Expertise	The government/industry team creates a richer pool of experience in perceiving and dealing with risks.
Broad-Based Buy-In	Risks and mitigation strategies are cooperatively determined by the government/industry team, so results are accepted by all. Second-guessing and after the fact criticism are eliminated.
Risk Consolidation	Knowledge of each other's risks gives decision-makers global perspective and highlight areas of common interest and concern.

Table 6-12. Team Risk Management Benefits [HIGUERA94]

6.4.2.2.2 Team Risk Management Model

The Team Risk Management Model is illustrated in Figure 6-12. Each function has a set of activities backed by processes, methods, and tools that encourage and enhance communications and teamwork. The TRM Model adds *initiate* and *team* to the SEI Risk Management Paradigm. *Initiate* and *team* are defined in Table 6-13.

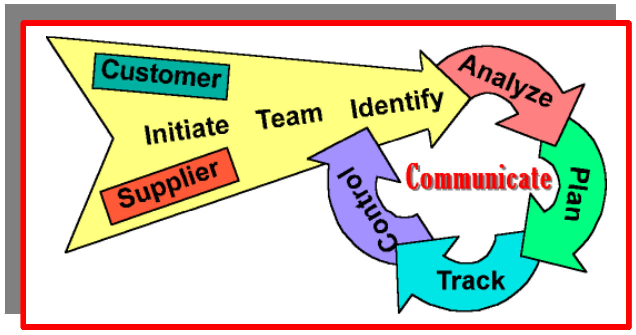


Figure 6-12. Team Risk Management Model

Function	Description
Initiate	Recognize the need and commit to creating the team culture. Either Government or industry may initiate team activity, but both must commit to sustaining the team.
Team	Formalize the government/industry team and merge viewpoints to form a shared product vision. Systematic methods establish a shared understanding of program risks and relative importance. Establish a joint database of risks, priorities, metrics, and action plans.
Identify	Search for and locate risks before they become problems. Identify risks and set program priorities to arrive at a joint understanding of what is important. Identify new risks and changes.
Analyze	Process risk data into decision-making information. Risk analysis is performed to determine what is important to the program, to set priorities, and to allocate resources. Group risks and quantify their impacts, likelihood, and time frame.
Plan	Translate risk information into decisions and mitigating actions (both present and future) and implement those actions. Establish mitigation plans for shared risks.
Track	Monitor risk indicators and mitigation plans. Indicators and trends provide information to activate plans and contingencies. Review them periodically to measure progress and identify new risks. Maintain visibility of risks, program priority, and mitigation plans.
Control	Correct for deviations from risk mitigation plans. Actions can lead to corrections in products or processes. Actions may lead to joint resolution. Changes to risks, risks that become problems or faulty plans require adjustments in plans or actions. Maintain a level of risk that is acceptable to the program managers.
Communicate	Provide information and feedback internal and external to the program on risk activities and current or emerging risks. Communication occurs formally as well as informally. Establish continuous, open communication. Formal communication about risks and action plans is integrated into existing technical interchange meetings, design reviews, and user requirements meetings.

Table 6-13. Team Risk Management Functions and Description

6.5 Applied Software Risk Management

When an effective risk management program is in place, planning can focus on avoiding future problems rather crisis management. Lessons learned can be applied to avoiding future crises rather placing blame. Work plan activities can be evaluated for their effect on overall program risk, as well as on schedule and cost. Important meeting agendas can be designed to discuss risks and their effects before discussing the specifics of technical approaches or current status. There is free flow of information among all program levels, because it is coordinated by a centralized system that captures identified risks and the information about how they are analyzed, planned, tracked, and controlled. This is achieved when risk is no longer treated as a four-letter word, but rather is used as a rallying mechanism to arouse creative efforts.

With effective risk management, potential problems are recognized and dealt with daily, before they occur. The finest software can be built within budget and on time. People, work groups, and projects throughout the program understand that they have a shared common vision to build a world class product on a program with a successful outcome. [WILLIAMS97] The following examples show that, when successfully applied, risk management is an investment in success.

6.6 Software Risk Management Begins with You!

"Risk management is quickly becoming a mature discipline. To achieve the promise of fully effective software risk management, the software industry still must address several continuing challenges.

- *Achieve commitment of all key stakeholders (developers, customers, users, maintainers, and others) to a risk management approach.*
- *Establish an evolving knowledge base of risk management experience and expertise, organized for easy and collaborative use by all stakeholders.*
- *Define and propagate mature guidelines on when and how to avoid, prevent, transfer, or accept and manage risk.*
- *Develop metrics and tools for reasoning about risk management's return-on- investment issues, including guidelines for deciding how much of a risk reduction activity is enough."* — Barry W. Boehm and Tom DeMarco [BOEHM97]

Software acquisition and management may be the greatest challenge of your career. Software programs are more prone to failure than success. The most important things you have to manage are the inherent risks associated with the development and support of all major software-intensive systems. Risk management encompasses a defined set of activities and mitigation techniques. How you implement these activities and techniques are defined in structured, disciplined risk management methodologies. The risk management methodology you choose must be tailored and selected based on risks peculiar to your program. It must also be systematic, repeatable, and based on solid, proven risk management techniques. A disciplined risk management approach is essential for program insight and decision-making on preventive actions critical to program success.

Significant risks must be assessed and their impacts quantified in terms of quality, cost, and schedule. Proactive risk management must be budgeted for and include abatement plans based on quantified risk impacts. Time and dollar requirements for risk management must be included in the total estimated conract cost.

Risk elements must be tracked throughout the acquisition life cycle. Techniques for managing risk are discussed throughout these Guidelines. They include the following software acquisition best practices.

- Software development maturity assessments;
- Software engineering discipline;
- Process, product, and program monitoring through measurement and metrics;
- Reuse;
- Continuous program planning;
- Reviews, audits, and peer inspections;
- Defect prevention, detection, and removal; and
- Constant process improvement.

Risk management, on an iterative tradeoff basis, is key. It is a proactive way for you to prevent problems and be prepared. It is an insurance policy. It is a way for you to be armed with solutions if an anticipated (or unforeseen) problem interrupts or negates your plans. The time, effort, and funds you dedicate to its practice will be well spent. Risk management is a necessary, sound investment in program success. Remember, *software risk management begins with you!*

6.7 References

[AUGUSTINE83] Augustine, Norman R., *Augustine's Laws*, American Institute of Aeronautics and Astronautics, New York, 1983.

[BLUM92] Blum, Bruce I., Software Engineering: A Holistic View, Oxford University Press, New York, 1992.

[BOEHM91] Boehm, Barry W., "Software Risk Management: Principles and Practices," *IEEE Software*, January 1991.

[BOEHM97] Boehm, Barry W., and Tom DeMarco, "Software Risk Management," *IEEE Software*, May-June 1997.

[CARR93] Carr, Marvin J., et al, Taxonomy-Based Risk Identification, CMU/SEI-93-TR-6, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, June 1993.

[CARR97] Carr, Marvin J., "Risk Management May Not Be for Everyone," *IEEE Software*, May-June 1997.

[CHARETTE89] Charette, Robert N., Software Engineering Risk Analysis and Management, McGraw-Hill Book Company, New York, 1989.

[CONROW97] Conrow, Edmund H., and Patricia S. Shishido, "Implementing Risk Management on Software Intensive Projects," *IEEE Software*, May/June 1997.

[DAU98] Defense Acquisition University, Risk Management Guide for DoD Acquisition, Defense Systems Management College, March 1998.

[EVANS94] Evans, "Thread of Failure: Project Trends That Impact Success and Productivity," *NewFocus*, Number 203, Software Program Managers Network, Naval Information System Management Center, March 1994.

[FAIRLEY94] Fairley, Richard, "Risk Management for Software Projects," *IEEE Software*, May 1994.

[HAIMES95] Haimes, Yacov Y., and Clyde Chittister, "An Acquisition Process for the Management of Nontechnical Risks Associated with Software Development," Acquisition Review Quarterly: The Journal of the Defense Acquisition University, Defense Systems Management College, Fort Belvoir, Virginia, Spring 1995.

[HALL97] Hall, Elaine M., Managing Risk: Methods for Software Systems Development, Addison Wesley Longman, Inc., Reading, Massachusetts, 1997.

[HENDERSON97] Henderson, Derek E., and Andrew P. Gabb, "Using Evolutionary Acquisition for the Procurement of Complex Systems," DSTO Electronics and Surveillance Research Laboratory, Salisbury, South Australia, March 1997; Defense Acquisition Deskbook, U. S. Department of Defense, The Pentagon, Washington, DC, 18 December 1998.

[HIGUERA94] Higuera, Ronald P., et al, Team Risk Management: A New Model for Customer-Supplier Relationships, CMU/SEI-94-SR-5, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, July 1994.

[HIGUERA96] Higuera, Ronald P., and Yacov Y. Haimes, Software Risk Management, CMU/SEI-96-TR-012, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, June 1996.

[LEE33] Lee, GEN Robert E., as quoted by J.F.C. Fuller, Grant and Lee: A Study in Personality and Generalship, Eyre and Spottiswoode, London, England, 1933.

[LISTER97] Lister, Tim, "Risk Management Is Project Management for Adults," *IEEE Software*, May-June 1997.

[MAURICE600AD] Maurice, Flavius Tiberius, The Strategikon, circa 600AD.

[McCONNELL97] McConnell, Steve, "Software's Ten Essentials: Prospecting for Programmers' Gold," *IEEE Software*, March-April 1997.

[MILLS95] Mills, Andy, "Software Acquisition Improvement: Streamlining Plus Risk Management," paper presented to the Seventh Software Technology Conference, Salt Lake City, Utah, April 1995.

[MOSEMANN95] Mosemann, Lloyd K., II, as quoted by Edmond H. Conrow and Patricia S. Shishito, "Implementing Risk Management on Software-Intensive Projects," *IEEE Software*, May-June 1997.

[NAPOLEON31] Napoleon Bonaparte I, The Military Maxims of Napoleon, 1831, David Chandler, editor, George C. D'Aguilar, translator, Greenhill Books, London, 1987.

[NAPOLEON55] Napoleon Bonaparte, as quoted by Christopher J. Herold, editor, The Mind of Napoleon: A Selection from His Written and Spoken Words, Columbia University Press, New York, New York, 1955.

[NAVY95] Taught in the Employment of Naval Forces course, "The Military Planning Process," U.S. Naval War College, Newport, Rhode Island, 1995.

[PRESSMAN93] Pressman, Roger S., "Understanding Software Engineering Practices: Required at SEI Level 2 Process Maturity," Software Engineering Training Series briefing presented to the Software Engineering Process Group, 20 July 1993.

[ROETZHEIM88] Roetzheim, William H., Structured Computer Project Management, Prentice Hall, Englewood Cliffs, New Jersey, 1988.

[SISTI94] Sisti, Frank J. and Sujoe Joseph, Software Risk Evaluation Method, Version 1.0, CMU/SEI-94-TR-19, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, December 1994.

[STORMONT95] Stormont, 1st Lt Daniel (USAF), "Risk Management for the B-1B Computer Upgrade," paper presented to the Seventh Software Technology Conference, Salt Lake City, Utah, April 1995.

[WILLIAMS97] Williams, Ray C., and Audrey J. Dorofee, "Putting Risk Management into Practice," *IEEE Software*, May-June 1997.